NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

A FORMAL SPECIFICATION AND ANALYSIS OF THE RESOURCE RESERVATION PROTOCOL

by

David P. Hensley

September 1999

Thesis Advisor: Gilbert M. Lundy

Approved for public release; distribution is unlimited.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY ***(Leave blank)***	**2. REPORT DATE** September 1999	**3. REPORT TYPE AND DATES COVERED** Master's Thesis
4. TITLE AND SUBTITLE : A Formal Specification And Analysis Of The Resource Reservation Protocol		**5. FUNDING NUMBERS**
6. AUTHOR(S) Hensley, David P.		
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000		**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A		**10. SPONSORING / MONITORING AGENCY REPORT NUMBER**
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.		
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.		**12b. DISTRIBUTION CODE**

13. ABSTRACT ***(maximum 200 words)***

This thesis explores the practicality of using the Resource ReSerVation Protocol (RSVP) model to prove quality of service guarantees over IP networks. An overview of the requirements to provide quality of service is provided. Using Finite State Machine analysis, the RSVP protocol is formally specified and found to be suitable for reserving resources along a proposed path. However, the distributed nature of the RSVP model and its reliance on quality of service aware routing protocols is problematic. Several examples where RSVP provides less than optimal and/or incorrect results are studied. The framework for alternate model of proving quality of service is proved. This model uses a centralized server for flow path computation. The server-based approach provides more accurate results than the RSVP model and is capable of network optimization; yet it places fewer strains on network resources and appears easier to implement.

14. SUBJECT TERMS Networking, IPv6, Internet Protocol, RSVP, Resource Reservation			**15. NUMBER OF PAGES** 72
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UL

NSN 7540-01-280-5500 **Standard Form 298 (Rev. 2-89)**
Prescribed by ANSI Std. 239-18

Approved for public release; distribution is unlimited

A FORMAL SPECIFICATION AND ANALYSIS OF THE RESOURCE RESERVATION PROTOCOL

David P. Hensley
Captain, United States Marine Corps
B.A., University of Oklahoma, 1990

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
September 1999

Author: [signature]
David P. Hensley

Approved by: [signature] 9/10/99
Gilbert M. Lundy, Advisor

[signature] 9/13/99
Wolfgang Baer, Second Reader

[signature]
Dan Boger, Chairman
Department of Computer Science

ABSTRACT

This thesis explores the practicality of using the Resource ReSerVation Protocol (RSVP) model to prove quality of service guarantees over IP networks. An overview of the requirements to provide quality of service is provided. Using Finite State Machine analysis, the RSVP protocol is formally specified and found to be suitable for reserving resources along a proposed path. However, the distributed nature of the RSVP model and its reliance on quality of service aware routing protocols is problematic. Several examples where RSVP provides less than optimal and/or incorrect results are studied. The framework for alternate model of proving quality of service is proved. This model uses a centralized server for flow path computation. The server-based approach provides more accurate results than the RSVP model and is capable of network optimization; yet it places fewer strains on network resources and appears easier to implement.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENT

The author would like to acknowledge Linvx and PC. They understand.

I. INTRODUCTION

A. MOTIVATION

A stumbling block to the successful integration of voice, video and other stream-oriented flows over Internet Protocol (IP) networks is the current best-effort model of IP routing. Under best-effort routing, each packet is treated equally with no consideration given to the needs of one packet over those of another. Additionally, traditional IP routing has no understanding of a flow, each packet stands alone to traverse the network resulting in bursty delivery of IP datagrams. This egalitarian approach gave satisfactory performance with early uses of the Internet, such as e-mail and network file services but became strained with later uses, such as HTTP. The slow packet delivery times, dropped packets and high jitter inherent in the current IP routing model is proving completely unsatisfactory for the needs of IP telephony, video and other stream-oriented applications. To be useful, packets that are part of the aforementioned real-time streams must arrive in order and the network must guarantee minimal levels of throughput and delay. This requires special handling. Throughput guarantees, known as quality of service are not engineered into the current IP routing model.

The problem of providing quality of service over IP is the subject of much current research within the Internet community. One approach to providing guaranteed quality of service over IP networks is the Resource ReSerVation Protocol (RSVP) in conjunction with modified routing protocols. RSVP is a soft-state receiver-oriented reservation setup

protocol. This model of quality of service uses RSVP to provide the setup mechanism for establishing and maintaining resource reservations within the sending and receiving hosts and all intermediate nodes. It utilizes the underlying routing protocol to determine the path of a flow. The flow path is established from the sender to the receiver, while the receiver makes the actual quality of service request. As the request travels from the receiver to the sender, each intermediate node learns the quality of service requirements, verifies the requester and then based upon available resources either grants or denies the request. The reservation is only valid for the duration of the flow. As long as the flow stays within the agreed parameters the network, barring link or hardware failures, guarantees minimum levels of throughput and latency. This model isolates the routing decisions from reservation setup mechanism. This RSVP approach was introduced in 1995 and is now a proposed Internet standard. It enjoys widespread support from both academia and industry. All of the major router vendors, Cisco, Bay Networks and 3Com, currently support RSVP in their production routers.

Providing integrated services over IP requires a fast, scalable and robust mechanism while minimizing network overhead. Since whichever resource reservation mechanism selected will provide the backbone for tomorrow's integrated IP networks, selection of the most appropriate method is essential.

B. OBJECTIVE

The objective of this thesis is to analyze the RSVP Protocol using a system of Finite State Machines and determine the feasibility of the model in providing quality of

service within an integrated services environment. A critique and suggestions for improvement are also provided.

C. THESIS OUTLINE

This thesis is divided into six chapters. Chapter II provides a notional model for quality of service, explaining services that any quality of service mechanism must provide. Chapter III is an overview of the next version of IP, IP version 6 (IPv6) and explains why its adoption is crucial to providing quality of service over IP. Chapter IV is a description, analysis and critique of RSVP protocol. Chapter V proposes a simpler, yet more powerful alternate model for providing quality of service over existing IP networks. Finally, Chapter VI draws conclusions and provides follow-on work to this thesis.

THIS PAGE INTENTIONALLY LEFT BLANK

II. QUALITY OF SERVICE

This chapter explains the origins and deficiencies of the best-effort routing model. The chapter then defines quality of service, describes its importance and provides a notional model a mechanism must use to provide guaranteed service levels in IP networks.

A. LIMITATIONS OF BEST EFFORT ROUTING

In the late 1960s, the Defense Advanced Research Projects Agency (DARPA) began research to develop robust data networks capable of surviving the loss of multiple nodes. DARPA's efforts led to development and adoption of an entire suite of protocols in the 1970s and early 1980s that provided a packet switched network where every packet contained the information necessary to traverse the network. Intermediate nodes, now called *routers*, are able to determine the best path to send a packet towards its destination. This model, called *best-effort routing* provides egalitarian treatment to all packets. All hosts are free to transmitted data on the network at will. The free-access policy causes over utilization of network resources leading to slow throughput, dropped packets and high jitter.

Although well suited for surviving nuclear warfare, best-effort routing's failure to recognize and provide special handling for different classes of traffic makes it unsatisfactory for providing integrated services. The Internet is no longer an esoteric network used by research institutions for the transmission of e-mail and file transfers.

Rather, it has evolved into a global communications network used for exchange of data, voice, video and real-time data. Applications and users require levels of service consummate with the throughput and delay needed for a given application. For example, streaming video requires high throughput, but may tolerate long delay, where an IP telephony conversation requires less bandwidth, but will not tolerate either excessive delay or jitter. Other applications, such as SMTP and FTP require no special handling and may traverse the network on a space available basis. Before successful widespread integration of telephone and video into IP networks, a scheme to provide quality of service with IP networks must be developed.

B. NOTIONAL QUALITY OF SERVICE MODEL

Every IP network consists of hosts and intermediate nodes called routers. A flow is defined as a stream of delay sensitive datagrams between a sending and receiving host across one or more routers that require minimal levels of throughput. Datagrams within a flow originate from the same host, and all require the same handling as they traverse the network towards their destination(s). Flows may be unicast to a single host or multicast to several hosts.

Quality of service to flows is achieved by reserving the network resources required for each flow and ensuring network resources are not over-subscribed. Network over-subscription results in congestion and results in low throughput and dropped packets. Scarce resources in IP networks include router computational resources and link capacity between nodes. Since routers know the capacity of each direct connected link, it

is possible for routers to make reservations and ensure the aggregate of the reservations does not exceed the capacity of the link. Router computational resources are preserved by selection of efficient routing models and algorithms.

Although the amount of transmission delay is a fixed property of the selected route, network throughput is variable. Routers control bandwidth allocation across links through queue manipulation. For example, if a given application requires one-half the available bandwidth of a link, the router provides that level of service by ensuring packets making up the flow, if present, have access to every other transmitted frame. This access is controlled by controlling the order of packets in the queue for that link. Actual queuing mechanisms utilized are beyond the scope of this thesis. Through queuing algorithms, IP networks may provide firm (mathematically provable) bounds on end-to-end queuing delays, which in turn guarantees a minimum limit of network throughput and delay. The guarantees also ensure flow packets will not be discarded due to queue overflows. [1]

This scheme does not eliminate jitter, but does reduce it. Jitter is reduced to the extent that quality of service places a maximum upper bound on the time packets take to traverse the network. However, the minimum bound is fixed by the propagation delay of the selected path. So, unless the network is one hundred percent utilized, many packets will arrive early. To overcome jitter, applications must provide buffering. Tests have shown that without network congestion, 800 byte buffers are sufficient to provide buffering for a coast-to-coast telephone conversion. [2] Another benefit of flow-oriented routing is ordered delivery of IP packets. Since datagrams all follow the same path, packets arrive in the order they were transmitted.

The first step in providing guaranteed quality of service is providing a mechanism for applications to request guaranteed minimum levels of service. The quality of service mechanism must provide APIs on each host and intermediate node for passing messages between the reservation process and other running processes.

Once a request is made, the reservation process must find a path through the network that has sufficient resources available to support the request. Path discovery is not a trivial issue and is similar in function to a routing protocol. Approaches range from simple trial and error to complex algorithms that find optimal paths. Ideally the quality of service mechanism will consider all combinations of existing flows and link properties to arrive at a solution that maximizes total system throughput. Finally, after the path is determined, the protocol must provide a mechanism for reserving resources at each router along the path. This is called resource reservation, and RSVP is one proposed method.

Once the route is established, the quality of service mechanism must react to network changes. If a route becomes unavailable, the mechanism must discover if another appropriate route exists, and if so, re-route the existing flow. Ideally, the mechanism will use idle time to discover better paths for network optimization.

After reservations are made, applications may transmit packets at will. To prevent rogue hosts from overwhelming network resources, the protocol must ensure that flows do not exceed the agreed upon service parameters. Packets not conforming to the approved request must either be dropped by the network or down graded to best-effort status.

The quality of service protocol must be able to identify, or classify, packets as part of unique flows. Each node must be able to discern flow packets and associate them with the appropriate quality of service parameters. Packets not associated with a flow are afforded best-effort routing. To ensure classification does not induce network latency, rapid packet classification is essential.

All of this must be accomplished while maintaining backward compatibility with existing IP networks. Specifically, the quality of service mechanism must maintain the ability to route via best effort, work with existing hardware and require minimal changes of current Internet protocols. Since rapid packet routing is the goal of IP networks, the impact of providing quality of service should be minimized on routers. Careful consideration should be made to ensure the selected model and related algorithms do not place undue additional computational and memory requirements on routers.

THIS PAGE INTENTIONALLY LEFT BLANK

III. ROLE OF IP VERSION 6

In this chapter the importance of IPv6 in providing quality of service is explained. Although many people see IPv6 as simply increased address space, it is much more. Using the current version of IP for quality of service is very difficult. The chapter explains why IPv6 features such as flow control, increased address space, reduced fragmentation and security make adoption an essential ingredient to providing quality of service.

A. BACKGROUND

The Internet Protocol is the most widely used network layer protocol in the world. IP is a connectionless packet delivery protocol that performs addressing, routing and control functions for transmitting and receiving datagrams over the network [3]. IP's related protocols, TCP and UDP, provide transport services. The basic IP unit is the datagram, with an IP header containing all the information needed to route the packet from its source to its destination. The current version of IP, version 4, was proposed in 1974 and adopted in 1984 as RFC 791. The exponential growth of the Internet along with the introduction of stream oriented applications have stretched the capabilities built into IPv4. Shortcomings include limited address space, lack of support for flows, explosive growth of routing tables and lack of native support for encryption or authentication. Although supplemental protocols have been developed and adopted to extend IPv4, the underlying protocol has not changed in twenty-four years. In 1994, with

the threat of address depletion looming, the Internet Engineering Task Force (IETF) recognized the need for a new version of IP and commissioned the IP Next Generation (IPng) working group. The IPng working group was charged to develop a new version of IP that addressed all the concerns with the previous version while maintaining compatibility with existing hardware, co-existing with IPv4 and requiring minimal modification of existing Internet standards and protocol. To that end, the working group was very successful.

B. IDENTIFICATION OF FLOWS

Although IPv6 is often touted as a fix to the problem of address depletion, increased address space is not its most important feature. The most important feature in IPv6 is the flow control label. In quality of service based networks, each intermediate node must examine each packet and determine if the packet is a member of a flow, and if so, associate it with the appropriate one. To properly classify a packet, the classifier examines various fields IP header of each packet. If any check fails, the packet is afforded best-effort service. Note that rapid elimination of packets is as important as the identification of flow-based packets. This classification must occur rapidly. While IPv4 provides no mechanism to assist this classification, IPv6's flow control label is designed for flow identification. The important role packet classification plays in quality of service, not address depletion, is the "killer app" which ensures adoption of IPv6.

1. Packet Classification in IPv4

The biggest problem with IPv4 vis-à-vis quality of service lies is the difficulty of classifying IPv4 packets. Identification of an IPv4 datagram as part of a unique data flow requires a router to examination three fields: the destination address, the IP protocol identification and the transport layer's destination port. Unfortunately, as seen in Figure 1, that information is spread throughout IPv4's variable-length IP and transport layer headers.

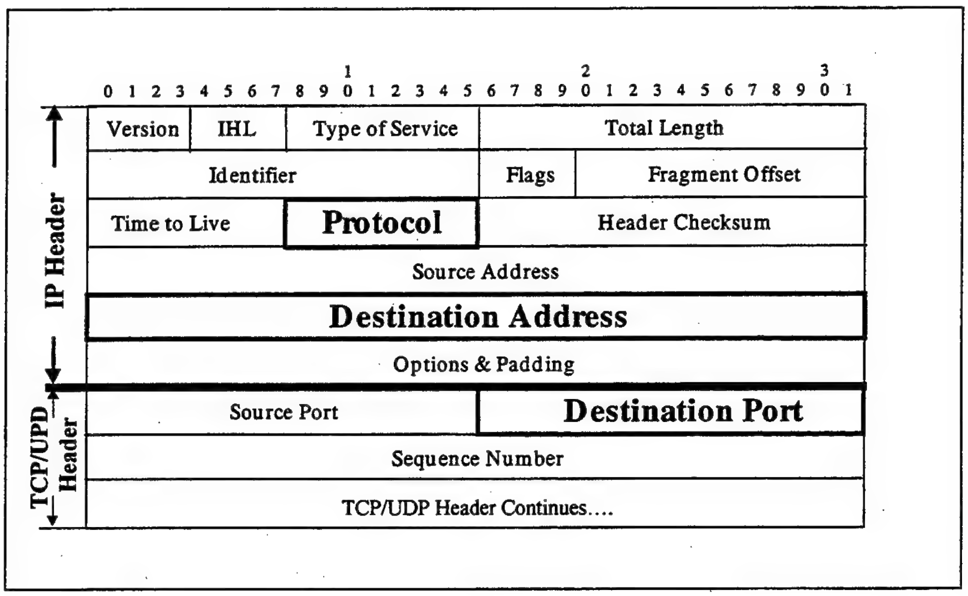

Figure 1: Three Components to Uniquely Identify IPV4 Flow

The classification process of IPv4 packets is complicated. First, the classifier looks at the destination IP address of the packet. If the IP address matches an entry in its quality of service tables, the classifier then examines the TCP/UDP port number. The transport layer's port number is examined second, since unique port numbers allows more

rapid elimination than examination of the IP protocol. However, this violation of the transport layer breaches the principle of abstraction between transportation layers. It also prevents encryption of the TCP/UPD headers and requires increased complexity in each router. Finally, if necessary, the classifier examines the IP protocol number and routes the packet accordingly.

If a packet is a member of a flow, IPv4 packet classification requires all three checks. Although one would think the transport protocol's port number coupled with the destination IP address would uniquely identify a flow, it must be kept in mind that UPD and TCP have the same range of port numbers. Thus, IPv4 packet classification requires every intermediate router to examine the entire IP header, plus the first 32 bits of the transport layer header. This means that each router must read at least 192 bits of the header of every packet in the network. The overhead is excessive and increases network latency.

2. Packet Classification in IPv6

IPv6 has many features to simplify packet classification. Via IPv6's flow label field, routers determine in the first 32 bits of the IP header whether an IP packet is part of a flow. As illustrated in Figure 2, the flow label field is twenty bits, uniquely identifying 2^{20} flows per sending host. When establishing a flow, each sending host randomly generates a unique flow label. This flow label is then used in every packet in the flow. A non-zero flow label notifies intermediate nodes that the packet is part of a flow. A look up of the flow label in the quality of service tables then determines the appropriate level

of service to provide. With over one million unique flow labels, it is unlikely a router will have two identical flow labels. If that occurs, the router will read the source address to uniquely classify the packet. If the flow label field is zero, the router knows immediately, without a table look-up, the packet is not part of a flow and provides best-effort service.

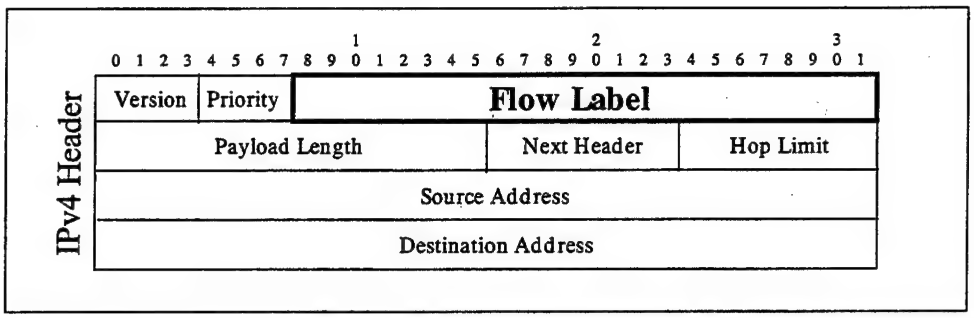

Figure 2: Flow Label for IPv6

Clearly, packet classification via the flow control label of IPv6 is much faster than under IPv4. This rapid classification of IPv6 packets is essential to providing quality of service of IP packets. The speed difference between classifying version four and version six packets is enormous, and as stated earlier, ensures the eventual adoption of IPv6.

C. INCREASED ADDRESS SPACE UNDER IPV6

Although IPv4's 32 bit addressing scheme provides over four billion unique addresses, they are rapidly being depleted. This is due to the rapid proliferation of networks coupled with the rigid and wasteful network/host division of IPv4 addresses. Clever patches such as Classless Inter-Domain Routing (CIDR), IP masquerading and

dynamic address assignment have reduced, but not eliminated, the need for additional IP address space. If, as predicted by the CEO of AT&T, the telephone (and by extension, FAX, pager, etceteras) become ubiquitous IP devices [4], the address space provided by IPv4 will be inadequate.

IPv6's 128-bit addressing scheme provides over 240,282,366,920,938,463,374, 607,431,768,211,456 addresses [3]. Although time has proven Bill Gates wrong when he predicted that 640k is more memory than any application will ever need, it appears safe to say that IPv6's 128 bit addressing scheme provides more than enough addresses for the future. Also assignment of IPv6 addresses is more efficient, because unlike the limitations imposed in IPv4, the entire address space of IPv6 is continuously bit-wise maskable.

1. Representation Of IPv6 Addresses

The preferred format for representing IPv6 addresses takes the form of X:X:X:X:X:X:X:X, where each X represents 16 bits in hexadecimal format [3]. The colons are used as separators. A sample IPv6 address is: FEDC:BA98:0:0:0:0:7654:800. To simplify representation of addresses, continuous zeros may be deleted and a double colon put in their place. Hence, the above address could be shortened to: FEDC:BA98::7654:800

2. IPv6 Addressing Hierarchy

IPv6 addresses are continuously bit-wise maskable. Administrators may determine, within their assigned range, which portion of the address is the network and

which portion belongs to the machine. At the extremes, the network portion of the address may be one bit (representing one giant network) or 126 bits (representing many networks of two machines each). The flexibility of selecting network boundaries encourages efficient addressing along logical boundaries. The designers of IPv6 have already divided the address space as indicated in following table.

Purpose	Prefix	% of Space
Reserved	0000 0000	1/256
Unassigned	0000 0001	1/256
NSAP Allocation	0000 001	1/128
IPX allocation	0000 010	1/128
Unassigned	0000 011	1/128
Unassigned	0000 1	1/32
Unassigned	0001	1/16
Unassigned	001	1/8
Provider Based Unicast Addresses	**010**	**1/8**
Unassigned	011	1/8
Geographic Based Unicast Addresses	**100**	**1/8**
Unassigned	101	1/8
Unassigned	110	1/8
Unassigned	1110	1/16
Unassigned	1111 0	1/32
Unassigned	1111 10	1/64
Unassigned	1111 110	1/128
Unassigned	1111 1110 0	1/512
Link Local Use	111 1110 10	1/1024
Site Local Use	1111 1110 11	1/1024
Multicast Address	**1111 1111**	**1/256**

Table 1: Allocation of IPv6 Addresses [3]

Close examination of Table 1 reveals that the designers of IPv6 have set aside one forth of the address space for unicast addresses. A small portion of the address space is

set aside to encompass the entire IPX and NSAP address ranges. It should also be noted that 1/256th of the address space, or 2^{124} addresses are reserved for multicasts. This is all accomplished with almost one-half of the available address space remaining unassigned and available for future use.

Although administrators are free to divide their address space as they see fit, Figure 3 illustrates the IPv6 designer's notional model of a unicast IP address. An example follows using Figure 3 and Table 1 as guides. Table 1 shows the IPv6 designers set aside 010:: as the space for unicast addresses, so the address must be within that range. Next, the IETF determines the appropriate amount of bits to set aside for top-level address registries. Right now there is only one registry, the Internic. However, future registries may be based along geographical boundaries such as continents or along political boundaries such as nations. Each registry would be assigned address space for their domain. This example uses the United States government as one of many registries. As illustrated, there may be 2^{m-3} registries. This number is expected to be very small, consuming at most eight bits.

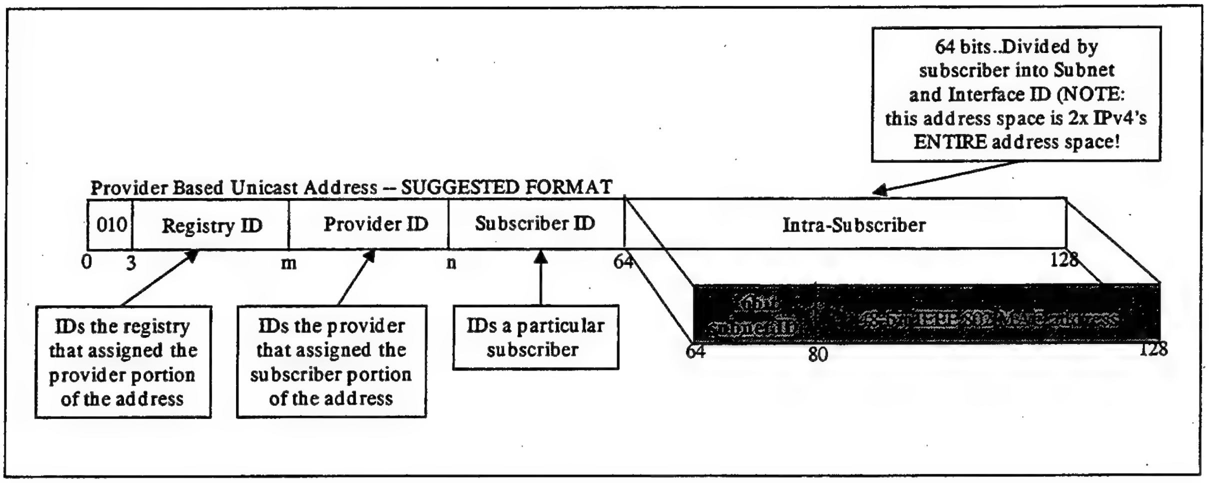

Figure 3: Suggested Format for IPv6 Unicast Label

In turn, each registry determines the appropriate number of bits to support the desired number of providers and sub-divides their address space along that boundary. The boundary set by the registry is represented by n and provides 2^{n-m} providers within that registry's domain. Each provider under the control of a registry gets its address from the registry's address space. In this example of the US government as a registry, then the Department of Defense would be a provider and would be assigned addresses from the US Government's address space.

As with the registries, each provider determines the appropriate number of subscribers to support and divides their address space accordingly. Continuing the example, a subscriber within the DoD would be the USMC. In the example illustrated in Figure 3, each subscriber then gets 2^{64} addresses, the minimum number of address space required for the auto configuration of hosts. If the subscriber wishes to use auto-configuration, 48 bits of that address will be used for the host portion of the address while 16 bits are used for the network portion. This provides 2^{16} networks with 2^{48} hosts

apiece. Although wasteful, it allows the IEEE MAC-layer address to be used for the host portion of the address, thus ensuring a unique host ID for every host on the Internet. If the subscriber does not use auto configuration, the 2^{64} addresses may be subdivided as needed.

D. REDUCED SIZE OF ROUTING TABLES

With IPv4, networks are assigned without regard to network topology. For example, adjacent Class C networks (or CIDR divided class B networks) that are adjacent in network space are almost never adjacent in network topology. This means the location of every network used in the Internet must be maintained in the routing tables of the Internet's backbone routers. As more IP networks were issued and existing networks were subdivided with CIDR, network routing tables exploded in size. The larger routing tables require additional RAM in routers and increase network latency due to increased route table look up time. At first glance, IPv6's entirely divisible address would seem to exacerbate the problem, careful design and implementation produce quite the opposite.

Since IPv6's unicast address space is provider or geographically oriented, the inner-division of a given provider's network need not be known outside that network. Within a provider's autonomous region, the interior routing protocol will have routes to every network within the provider's space. Thus, instead of advertising each individual network within a provider's cloud, routers on the edge of the provider's network will only advertise the higher level aggregate network to external routers. If a subscriber connects a lateral link to the external network, whatever aggregate network is contained within the

subscriber's interior routing protocol will be promulgated via that link. If the subscriber is running an interior routing protocol with the provider, then the entire provider's space will be advertised via that link. If the subscriber is running its own autonomous system, only the subscriber's network will be advertised.

Fragmenting IPv6 address spaces across non-contiguous regions will cause a huge explosion in routing tables. Although users are encouraged to use the address space of their provider, nothing in the standard prevents such fragmentation. However, the algorithms used in routing protocols provide a strong incentive to ensure routes advertised outside an autonomous region are as large as possible. As routers run out of space for their tables, entries are deleted from least significant to most significant. In other words, since a smaller address space represents fewer users than a larger space, when a router is forced to drop a table entry, it will always drop the one representing the fewer number of users. So, if a subscriber takes space from one provider and utilizes it to connect to another provider, this route may not be maintained in backbone routers.

E. OTHER FEATURES TO SPEED NETWORK THROUGHPUT

Other features included in IPv6 to speed the routing of IP datagrams are discussed in the following sections.

1. Fragmentation in Intermediate Nodes is Forbidden

Under IPv4, routers constantly fragment and reassemble packets for transmission across different links. This induces significant overhead in routers. IPv6 prohibits packet fragmentation at intermediate nodes. All packet fragmentation and reassembly must be

conducted at the communicating hosts. To facilitate this rule, IPv6 guarantees a Maximum Transmission Unit (MTU) of 576 bytes. Hosts may use the default maximum size or if some larger packet size is desired, applications may use MTU discovery to discover if larger packets are possible. Either way, relieving routers of packet fragmentation and reassembly increases network throughput.

2. Simplified Fixed Length Headers

The IPv4 header is variable length and contains variable-length optional headers. To ensure it does not miss any important information in an IPv4 packet, each router must examine the entire IP header, to include ALL optional headers. In contrast, IPv6 headers are of fixed length and contain extension headers. Unlike IPv4, information required by intermediate routers is contained only in the main header and the optional hop-by-hop extension header. The hop-by-hop extension header, if present, is required to immediately follow the main header. The presence of the hop-by-hop header is advertised in the next header field shown in Figure 2. This simplification in the analysis of IP headers reduces the latency of IPv6 networks.

C. SECURITY FEATURES OF IPV6

With the widespread adoption of IP telephony and e-commerce, IP must provide a mechanism to guarantee privacy and authenticate users. IPv6 provides native support via extension headers for per-packet encryption and authentication. Although used for e-commerce, these security features are also required to authenticate users requesting the

reservation of resources. Without authentication, hackers could conduct denial of service attacks or hijack resources on quality of service networks.

1. Packet Authentication

Whether used to ensure non-repudiation of an e-commerce transaction or to ensure the RSVP request is from a valid source, packet authentication is crucial. IPv6, via the authentication header provides data origin authentication and protection against replay attacks. Although the authentication header is cipher independent, all IPv6 implementations are required to support, at a minimum, the use of IP authentication headers with keyed Hashing for Message Authentication Codes (HMAC) with MD5 [3].

2. Packet Encryption

With the widespread availability of easy to use, yet sophisticated network sniffers, strong, per packet encryption is necessary. This is true especially in broadcast topologies. For example, without packet level encryption, cable modem users within the same collision domain are able to reconstruct their neighbors web sessions or read their e-mail. To protect privacy, IPv6 provides an extension header for Encapsulating Security Payload. Like the authentication header, the ESP header is cipher independent. However, all IPv6 implementations must support, as a minimum, the Data Encryption Standard (DES) in CBC mode. [3]

THIS PAGE INTENTIONALLY LEFT BLANK

IV. THE RESOURCE RESERVATION PROTOCOL (RSVP)

This chapter is a detailed specification and analysis of the RSVP protocol. First, the chapter provides an overview of the RSVP process followed by an explanation of the portions of the RSVP process. Next, detailed analysis of the RSVP process within the sender, receiver(s) and intermediate host(s) is made using a system of finite state machines. Finally, the chapter provides two examples where the RSVP model provides erroneous or less than optimal results.

A. DESCRIPTION

As discussed earlier, any resource reservation mechanism must address the following issues: finding a route that supports reservations and has sufficient unreserved capacity, the ability to adapt to a route failure and the ability to adapt to a new route change without failure. [5] RSVP is the proposed Internet standard for resource reservation over IP networks. RSVP is neither a transportation nor a routing protocol. Rather, it makes resource reservations along a path determined by the routing protocol for follow-on TCP or UDP data flows. Like the Internet Control Message Protocol (ICMP) or the Internet Group Management Protocol (IGMP), RSVP works alongside TCP/UDP. RSVP requests are simplex, hence an IP telephony phone call requires two RSVP setups, one in each direction. RSVP supports reservations for both unicast and multicast flows.

B. SPECIFICATION

1. Overview of a RSVP Request

As shown in Figure 4, the RSVP process is initiated outside of the RSVP protocol with the receiving host notifying an application on the sending host it wishes to receive information. In the case of an application such as Real Audio, this is accomplished via IGMP. If the application requires guaranteed quality of service for its datagrams, it uses a standard application programming interface to notify the RSVP process of the required Quality of service parameters and IP address of the destination host. The RSVP process on the sending host generates a raw IP datagram called a PATH message. This PATH message serves two purposes. First, it establishes a "path" for the flow. Secondly, the PATH message carries a suggested quality of service parameters to be delivered to the receiver(s). These parameters are embedded in a flowspec object. The flowspec defines both the quality of service and the identification of the set of packets that make up the flow. The RSVP process addresses the path message to the receiving host, encloses the flowspec object and transmits it onto the IP network. Path messages are routed through the network like normal IP packets, with the route determined by the network's routing protocol. RSVP messages are assigned the IP protocol number 46. This allows each RSVP enabled host to identify it as a RSVP packet and pass it to the RSVP process for further examination and possible action. To establish the path for the reservation, each intermediate node records the IP address of the previous host, replaces the IP address in the previous host field with its own and forwards the packet to the next hop towards the

destination. This process is repeated until the PATH message is received at the receiving host.

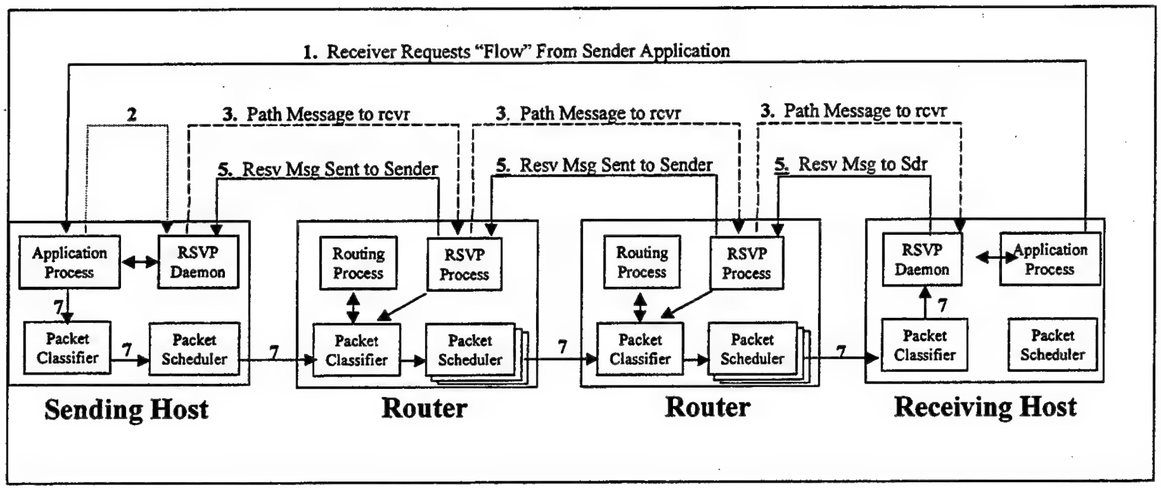

Figure 4: Overview of the RSVP Process

Upon receipt, the IP process on the receiver passes the PATH message to the RSVP process. In turn the RSVP process sends the Quality of service parameters contained in the flowspec to the appropriate application. The application decides the level of quality of service to request, and sends the final parameters to the RSVP process via a well-defined programming interface. The RSVP process takes the parameters, generates a flowspec object and embeds it in a RESV message. To ensure it follows the path established by the PATH message, the RESV message is addressed not to the sending machine, but to the machine identified as the previous hop when the PATH message was received. This ensures the RESV traverses hop-by-hop the route established by the PATH message.

Each intermediate node examines the RESV request and, if it grants the level of service requested, configures itself and forwards the packet to the next router in the path. Once the RESV message reaches the sending host, the RSVP process notifies the application of the establishment of the resource reservation. The application begins sending datagrams in accordance with the flowspec parameters. Note that although the sending application generated a proposed level of Quality of service, the receiving application determined the actual level of resources to reserve. This receiver-based reservation is a tenant of RSVP.

To keep the flow active and if necessary to discover alternate routes, the sending host periodically sends PATH messages to each receiver. The process continues in parallel to the dataflow until the resource reservation is terminated. RSVP reservations may be terminated two ways; timeouts at any portion of the RSVP path or a termination request from either the sender or receiver.

If the sender terminates the reservation, the sender's RSVP process generates a PATH_TEAR message. The PATH_TEAR message is forwarded along every branch of the tree, destroying all reserved resources along the way. If terminated by the receiver, a RESV_TEAR message is generated and sent along the path until it reaches a branch where it will not have any affect. A RESV_TEAR message not forwarded past a given point implies the receiver was part of a multicast tree and the path joined another receiver's path which made a reservation equal to or greater than the one being cancelled. By definition, all unicast RESV_TEAR messages reach the sender.

If the reservation is terminated by a timeout on an intermediate node, the node generates a PATH_TEAR message to send towards the receiver and a RESV_TEAR to send towards the sender.

2. Parts of the RSVP process

Figure 5 illustrates the RSVP process within an intermediate node. Shown are the packet classifier, packet scheduler, routing process and the RSVP process with its policy and administrative control modules. The RSVP message process begins with the examination of all arriving packets by the packet classifier.

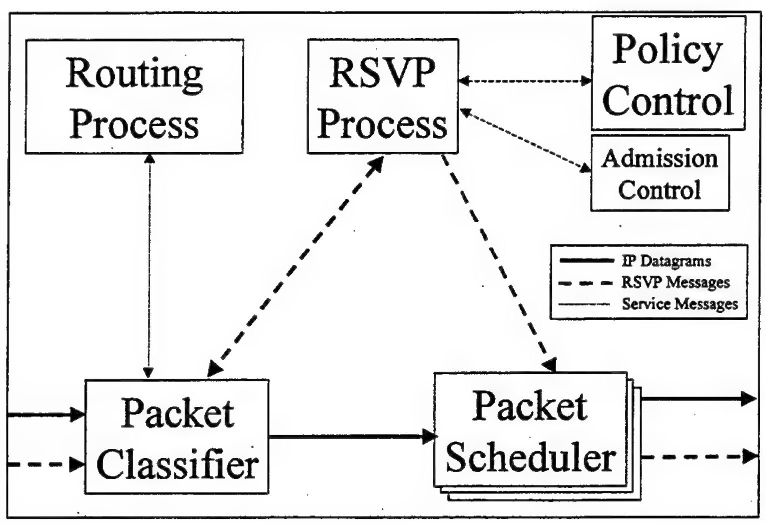

Figure 5: Anatomy of an RSVP Process

The classifier is responsible for determining the level of service required by each inbound packet. Since the classifier must examine every packet arriving at a node, rapid classification is essential. With respect to RSVP, the classifier may take three actions. If

the packet is determined not to be part of a flow, it is queued for best-effort service. If the network is congested, these packets may be dropped. If the packet is part of a flow, the classifier will determine via a lookup in the RSVP tables the required level of service and will queue it accordingly in the packet scheduler. Since it is impossible for the RSVP process to over-commit resources, these packets will only be dropped in the case of a link or hardware error. If the packet has an IP protocol ID of 46, the classifier identifies it as part of a RSVP service message and passes it to the RSVP process.

The RSVP process is responsible for granting RSVP requests and maintaining RSVP states. When the RSVP process receives a request for resources, it first passes the request to the policy control module to ensure the user has permission to make the reservation. [6] If the user has the authority to make the request, RSVP uses the admission control module to determine whether or not the router has sufficient remaining resources to grant the request. If the answer to either check is negative, the RSVP request is rejected. If both checks are approved, the RSVP process sets the appropriate parameters in the classifier and forwards the request to the next intermediate node. The packet schedule is a hardware device is the link-layer device, usually a port and associated queue(s). RSVP does not directly interface with the packet scheduler. However, quality of service is delivered by the packet classifier correctly manipulating the scheduler's queues for each interface.

3. Analysis of the RSVP Process Within a Sending Machine

The RSVP process on the sending machine (Figure 6) begins with RSVP in the idle state awaiting a RSVP request from an application. Upon receipt of a request containing the suggested quality of service parameters and destination address (either IP address or broadcast ID), the RSVP process examines the request for errors. If errors are found, the process rejects the message, notifies the application and returns to the idle state. If the parameters are acceptable, the RSVP process builds a properly formatted PATH message containing a flowspec object, addresses it to the receiver and transmits the datagram. After sending the PATH message, the process enters a wait state. If a RESV message is not received within the time-out period, another path message is generated and sent. If a RESV message is in not received after the allotted number of timeouts, the process sends a PATH_TEAR message, notifies the application and returns to idle.

After receiving a RESV message in reply to a path message, the RSVP process verifies the format of the flowspec. If correct, sends RSVP established messages to the receiver's RSVP process and notifies the application of the final reserved flowspec parameters and sends a RESV_ESTAB message to the receiver. The RSVP process enters the RSVP established stage while the application begins sending datagrams within the flowspec parameters. Periodically, the RSVP process will send PATH messages to refresh the reservation. Refresh PATH messages and identical in format and function to the original PATH message, and as shown in the figure are handled similarly.

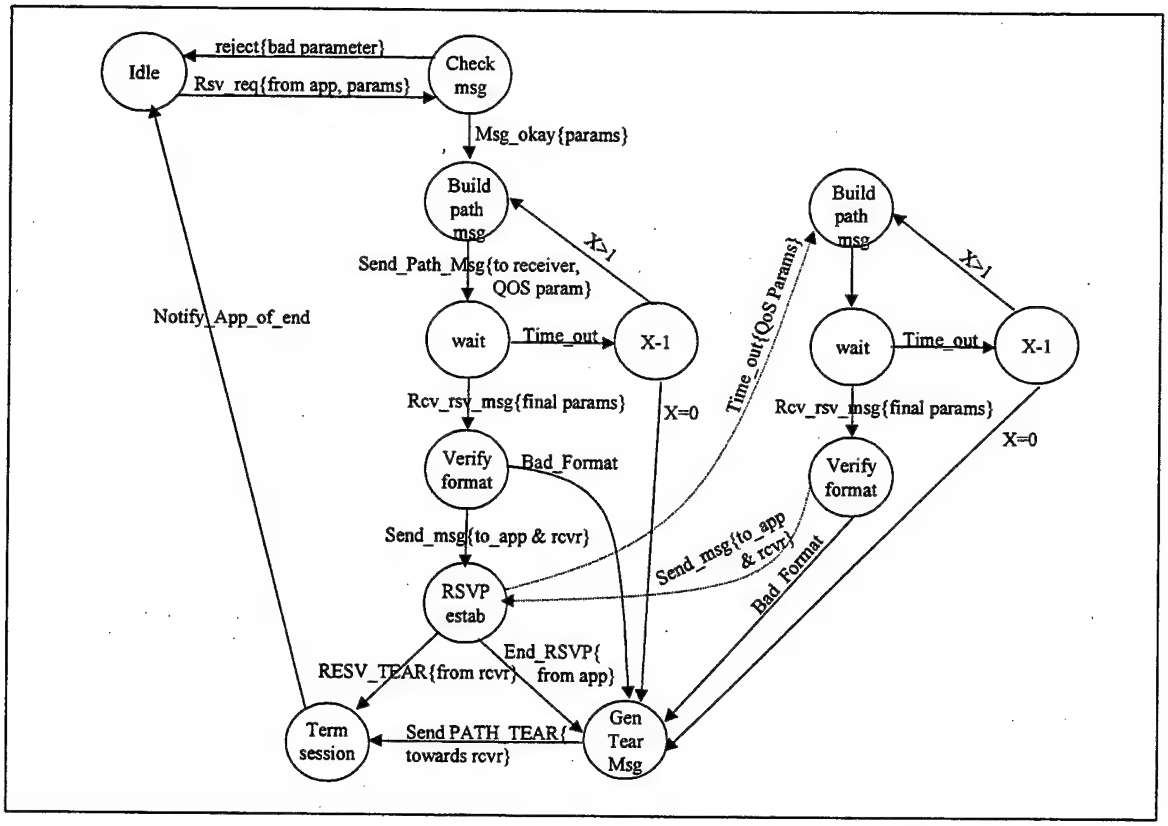

Figure 6: Sending Host RSVP Process

When notified by an application to terminate a reservation, the RSVP process terminates the session by generating and sending a PATH_TEAR RSVP service message towards the receiver. Once sent, the RSVP process notifies the application and returns to the idle state. Upon receipt of a RESV_TEAR message, the process notifies the application and returns to the idle state.

4. Analysis of the RSVP Process Within a Receiving Machine

As shown in Figure 7, the receiving machine starts idle waiting to receive a PATH message. Upon receiving a PATH message, the RSVP process checks the message for

errors. If errors are found, the process drops the message and returns to the idle state. If the message is good, the process sends the flowspec objects to the appropriate application and waits for a response. If a response is not received from the application or the request is rejected by the application, the message is dropped and the process returns to idle. Otherwise the RSVP process receives an updated flowspec object from the application, verifies the format and generates a RESV message. The RESV message is addressed to the last router in the PATH chain and sent onto the IP network. Once the RESV message is sent, the RSVP process enters a wait state.

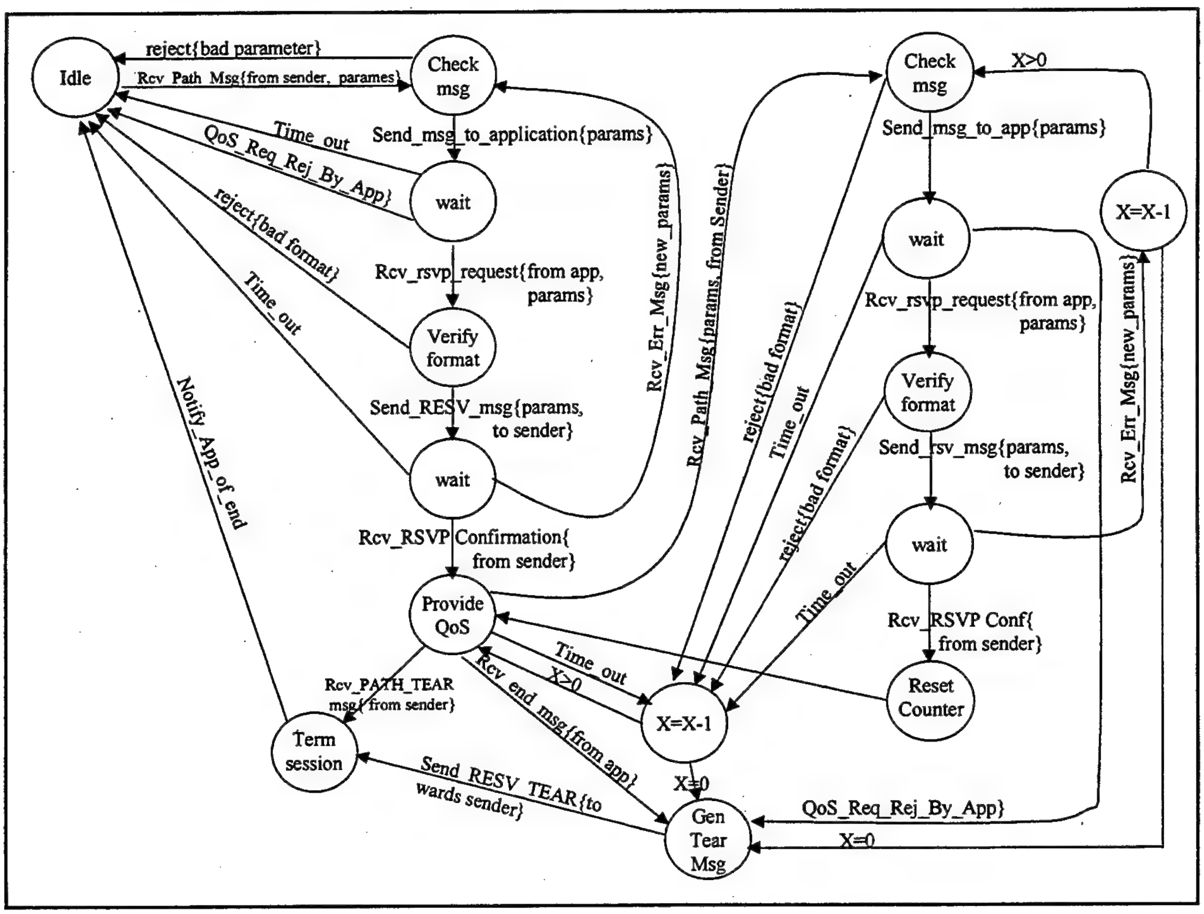

Figure 7: FSM of Receiving Machine

While in the wait state, three actions may occur. The process may not receive a response, in which case it will time out and return to an idle state. The process may receive a RSVP error message from any router along the path. In this case, RSVP will examine the message and forward it to the application. Finally, RSVP may receive a RESV_ESTAB message from the sending host acknowledging successful reservation. Once an acknowledgement is received, the RSVP process enters the provide provide QOS state.

Once the flow is established, RSVP looks for periodic PATH messages to maintain the established reservation. If it does not receive a new PATH message within X time periods, RSVP tears down the reservation by sending a RESV_TEAR message towards the sender. This happens regardless of the status of the on-going flow. PATH refresh messages are treated similarly to the initial path message. Close examination reveals the steps in processing the refresh message are identical to those of the initial PATH message. Differences arise in the results of each action. For example, when receiving a RCV_ERR message, the RSVP process must first ensure the reservation timer (X) has not expired prior to forwarding the message.

If the RSVP process receives a PATH_TEAR message, it immediately notifies the application and returns to the idle state. If the receiver must terminate the reservation, through either a timeout or a termination notification from the application, the RSVP process generates a RSVP_TEAR service message and transmits it via the RSVP path. Immediately after transmission of the RESV tear down message, RSVP will notifies the application and returns to the idle state.

5. Analysis of RSVP Request Within an Intermediate Host

The FSM analysis of the RSVP process at intermediate nodes is more complicated than that of RSVP hosts. Much of the increased complexity is due to cases presented by multicast flows. Like in the other FSMs, intermediate host processes begin in an idle state, waiting for PATH messages. Upon receipt of a PATH message, the RSVP process checks the format. If the format is bad, the packet is dropped and the process returns to idle state. Otherwise, the RSVP process stores the IP address identified in the PREV_ADDRESS field of the RSVP header, replaces the address with its own and forwards the PATH message to next hop towards the receiver. After forwarding the PATH message, RSVP waits for a return RESV message. If, after some time interval, a RESV message is not received, or upon receipt of a PATH_TEAR message, the process returns to idle.

Once a RESV message is received, RSVP checks the format of the message and verifies, via policy control, whether the user is authorized to make the resource reservation. If either check fails, an error message is sent to the receiver and the session is terminated. If the message passes those checks, administrative control is checked to verify the router has sufficient resources available to grant the request. If insufficient resources remain to grant the request, RSVP keeps the request open, builds an error message containing acceptable alternate Quality of service parameters and sends the revised flowspec proposal to the receiving host. If the Quality of service request is granted, the new parameters are compared with existing parameters (if any) and parameters are set in the router's packet classifier. The timeout counter is reset during

this step. If the reservation is a new request, or it modifies the parameters of an existing multicast flow, a RESV message is generated and sent to the next router up the path chain. After the message is sent, the process enters the provide_QOS state. Note that if the request is merged into an existing flow, no RESV message is generated or forwarded.

While in the provide_QOS state, RSVP may timeout, receive TEAR messages or receive PATH refresh messages. Time out is the simplest case. If after X timeout periods, the RSVP process has not received a RESV message, the process generates and sends tear messages and returns to idle. If a REC_TEAR message is received, the process must check to see if the receiving host is part of a multi-cast group. If it is the last receiver associated with the flow, a teardown message is generated and sent prior to returning to the idle state. If the receiver dropping its request is part of a multicast group, the process must determine if dropping the receiving host affects the aggregate upstream quality of service request. If dropping the receiver results in no change of parameters for the flow, the process returns to the provide_QOS state. On the other hand, if dropping the receiver results in a net change in Quality of service parameters, a RESV message containing the new flowspecs is generated and sent upstream.

If a PATH_TEAR message is received, the RSVP process must ensure the message is forwarded towards all receivers of that flow. The process will repetitively send path messages to each receiver then return to the idle state.

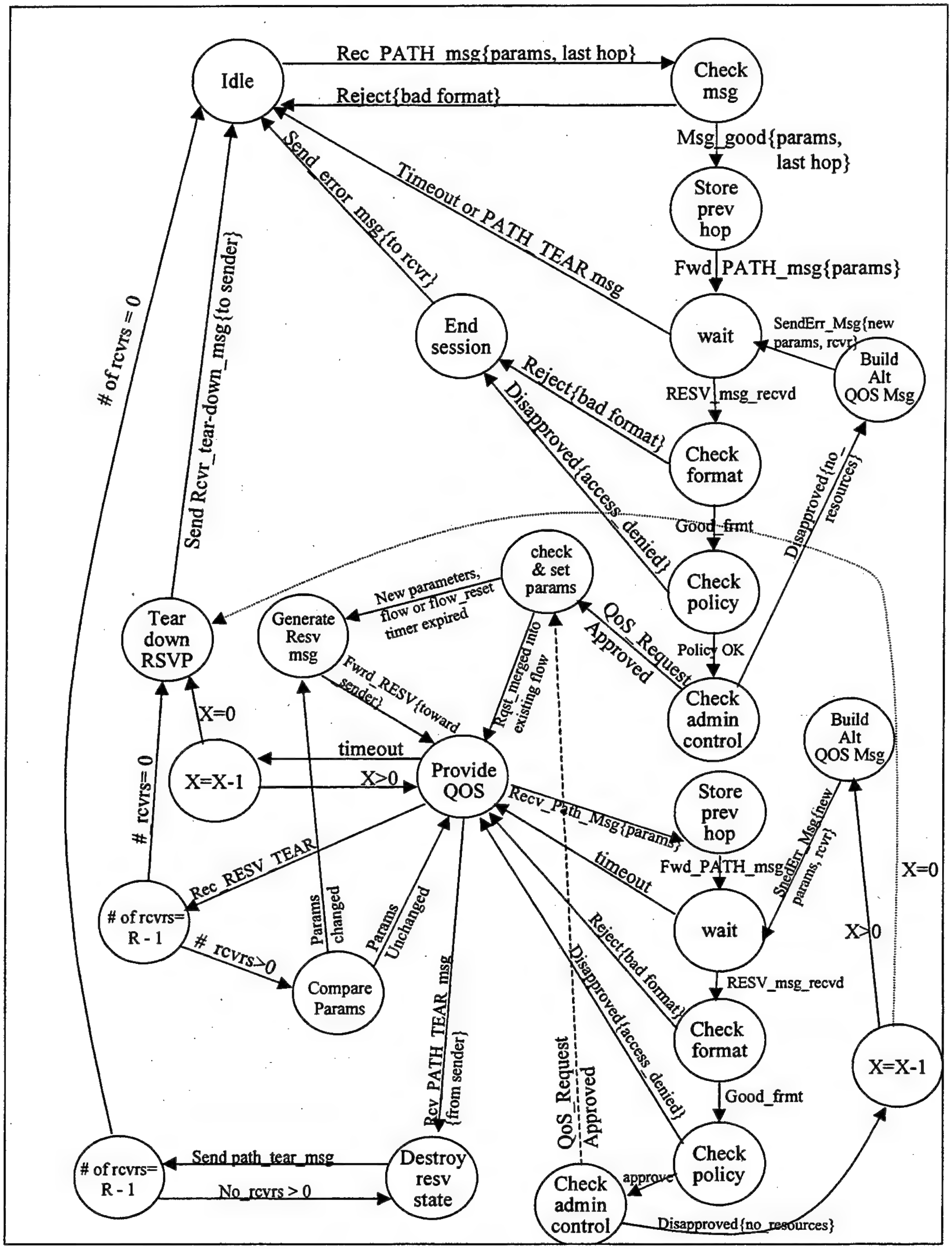

Figure 8: FSM of Intermediate Node RSVP Process

Finally, while providing Quality of service, the RSVP process may receive an updated PATH message. PATH refresh messages are identical to and are processed similarly to the initial PATH messages. Differences occur in error handling of PATH refresh messages. As seen in the figure, instead of returning to IDLE after an error, the process generally returns to the Provide_QOS state. If a corresponding RESV message is received and approved, the process then sets the appropriate parameters and returns to provide_QOS.

C. CRITIQUE

As a reservation mechanism, RSVP performs adequately. However, the model of separating the reservation mechanism from the underlying routing protocol is problematic. The decentralized model of providing quality of service, where each router makes decisions of the appropriate path for the flow presents several instances where requests for resources are denied when resources to meet them exist. Also, successful implementations of RSVP require each RSVP enabled router to a quality of service based routing protocol, Open Shortest Path First with quality of service extensions (QOSPF). As illustrated in the following examples, quality of service aware routing protocols do not ensure error free reservations. However, they do create undesirable side-effects such as larger routing tables, more frequent exchange of tables and increased demands on router computational resources. All are incongruent with faster networks. The following example illustrates a flaw in the RSVP model.

1. RSVP Denies Service When Alternate Path for Service Exists

In the scenario illustrated in Figure 9, the routing protocol routes the PATH message from the sending host to the receiving host via R1, R2 and R3. As previously discussed the RESV message must traverse the reverse of the route taken by the PATH message. Imagine the receiver requests 64kbps of throughput. R3 grants and forwards the request to R2. Since the link between R1 and R2 does not have sufficient resources remaining to honor the request, it is rejected at R2. However, examination of the figure shows that sufficient resources exist to grant the request over the path defined by the path R3->R4->R5->R1. However, RSVP will not even consider this path.

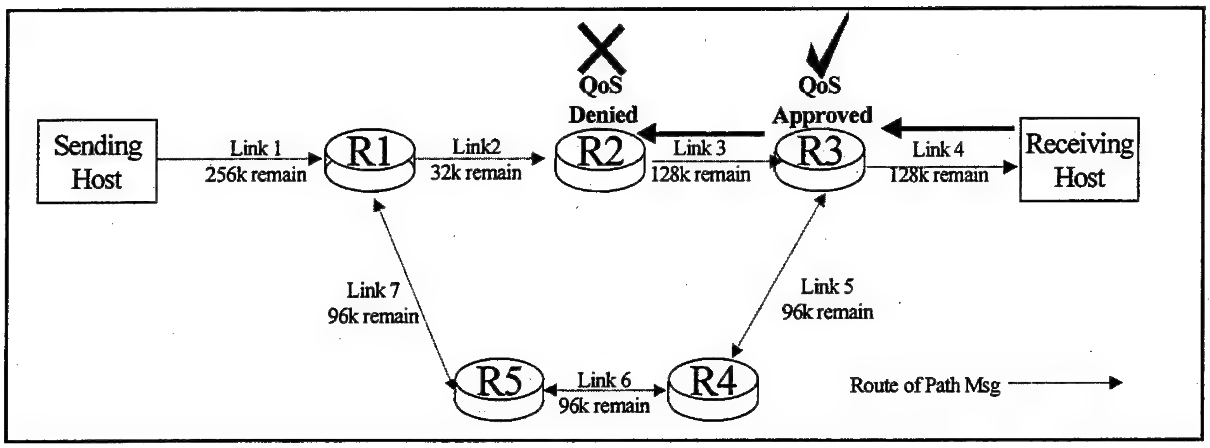

Figure 9: RSVP Rejects Request Resources Exists on an Alternate Path

RSVP designers acknowledge this problem and consider it a routing protocol issue. The designers point out that when used with a quality of service based routing protocols, a PATH message will travel a route containing resources necessary to grant the

request. This explanation does not withstand scrutiny. First, since the receiver requests the quality of service parameters, the parameters contained in the PATH message, if any, are guesses. According to the protocol specification, the flowspec in the PATH message may be blank or contain a "default" minimum requirement. Even the most advanced routing protocols are unable to select an appropriate path for a flow if the flowspec does not contain the parameters required to determine the path. Secondly, even if the routing protocol selects a path containing enough resources to grant this request, timing issues may prevent a successful reservation. Sufficient resources may exist when the PATH message is received, but may be consumed prior to the return of the corresponding RESV message. The probability of this occurring increases as the load on the network grows. These problems illustrate the need for tight coupling between the reservation setup mechanism and the underlying protocol.

2. RSVP Denies Requests When Aggregate Remaining Resources Exist

If a path supporting the requested resources does not exist, all RSVP requests will be rejected even if aggregate network throughput exists. This problem becomes aggrevated as network utilization increases. A simple example is illustrated in Figure 10 where a high fidelity 1.54mb/s VTC link is requested between two hosts. The path message is sent via R1 and R2. When the receiver sends the RESV message, R2 rejects the request because the link between R1 and R2 does not have enough remaining capacity. If however, the two 64kbps flows currently running over link 2 were re-routed

via links 4, 5 and 6, the network would have ample capacity for all three flows concurrently.

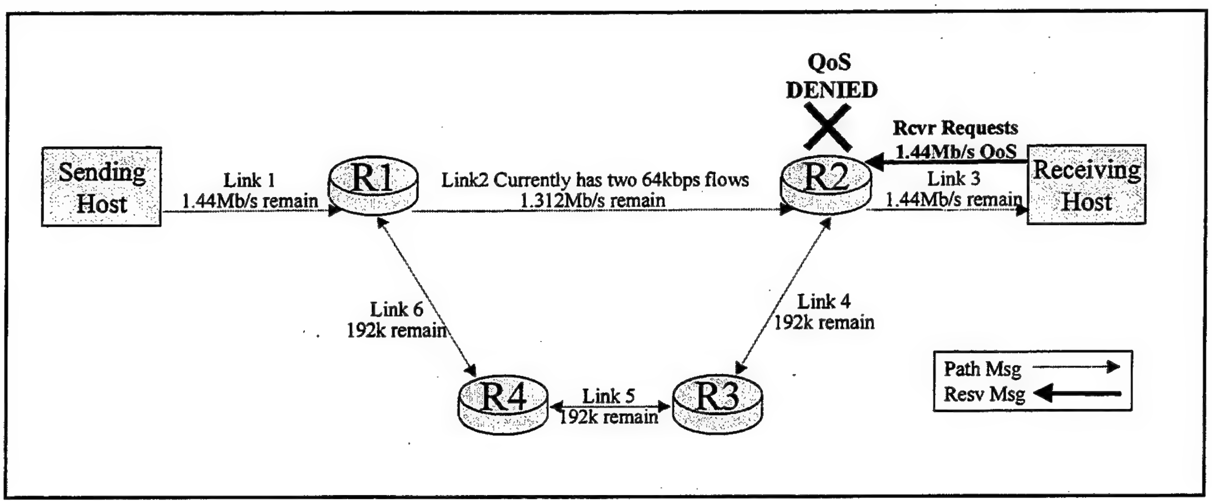

Figure 10: RSVP Rejects Request When Resources Exists Within Network

This example shows the need for a global reservation mechanism with the ability to reroute existing flows for network optimization. RSVP's distributed decision model coupled with the computational complexity of computing optimal paths prevents RSVP from optimizing network throughput.

This Bellman-Ford algorithm used in QOSPF finds the best path within the confines of existing flows, where "best" is defined as the path with the most available remaining throughput. If two routes are equal, the one with the fewest hops is selected. [7]. The complexity of the algorithm is O(RL), where R equals the number of routers and L represents the number of links. Since information about existing flows is not exchanged as part of QOSPF updates, it is impossible for the routing algorithm to

consider rerouting flows. In fact, the designers of QOSPF recognized the computational difficulty in computing optimal paths and since these calculations are performed at each router, they traded optimization for computational simplicity. Algorithms that compute optimal network utilization are possible, but their computational demands would significantly degrade router performance. However, if quality of service path computations are performed an a separate server vice each router, computational complexity becomes much less important.

In addition to the increased complexity of quality of service routing algorithms, routing protocols will require more information about each RSVP enabled routers. At a minimum, in additional to all the information currently required each router making a path decision must about resources remaining on each router. If network optimization is desired, information about the requirements of each existing flow must be exchanged. This information requires larger routing table exchanges, using up network bandwidth. Also, larger tables at each router will use consume RAM and slow table queries. Obviously, real-time information is ideal while computing a path, but constant resource updates are not desirable due to the overhead they would make on the system. Since real-time updates are not possible, by definition, any attempt to compute paths for quality of service in a distributed environment are inaccurate, for they are based on dated information. Dynamic networks would have to either accept incorrect paths or significantly increase the intervals between routing table updates. If accurate and timely path calculations are desired, the only solution to this dilemma is network-wide computation of routes on a single machine.

Also, in a distributed model, care must be taken to ensure routers are not constantly reconfiguring flows. Route thrashing could occur when a router computes an optimal path and re-routes a flow only to have a separate router re-route it yet again. This instability is inherent in distributed path computations and is unacceptable.

THIS PAGE INTENTIONALLY LEFT BLANK

V. AN ALTERNATIVE MODEL

This chapter proposes an alternate model to RSVP. As illustrated in the previous chapter, the distributed path calculation model used in RSVP wastes router resources and often provides incorrect results. The centralized quality of service model outlined in this chapter overcomes RSVP's shortcomings while providing all of its services.

A. OVERVIEW

The prior examples illustrate shortcomings of the RSVP model. Any quality of service mechanism that relies on hop-by-hop optimal path calculation at each intermediate node increases the computational demands on those routers and cannot be guaranteed to deliver correct results. The increased computational demands are due to RSVP's reliance on a quality of service aware routing protocol such as QOSPF. These advanced routing protocols rob computing power from every router in the network to compute the more complex algorithms and maintain the larger routing tables. Since the decisions are made hop-by-hop, each router must recalculate the optimal path algorithm.

Even with the most advanced path discovery mechanism, distributed route computation never guarantees a correct answer. Since routing table exchanges cannot be instantaneous, every machine is using stale link-state information in its calculations. Suppose a machine started optimal path calculations with fresh link state information. By the time the machine finished its calculations, link states may have changed. Also final

quality of service parameters are not present when the routing protocol determines the route of a the flow. Therefore, route determination under RSVP is a guess.

The answer to this dilemma is a centralized path calculation. A server used exclusively for the central determination of flow paths offers many advantages over the RSVP model. A centralized server will not suffer from any of the timing issues suffered by RSVP. Since it alone commits the network to new flows, barring hardware failure, it can have real-time information concerning flows for its calculations. Also, since its calculations do not rob processing power from routers, a server can use more complex algorithms arrive at more optimal solutions. This server based, system wide method for route determinations offers many other compelling advantageous over RSVP. The concept is illustrated in Figure 11.

1. Overview of the Server-Based Approach

This approach allows path decisions to be made in context of the entire network and in light of all existing flows. Under this approach a receiver notifies the server of the flow parameters required and the IP address of the sending host. The server maintains information about all network links and their capabilities, all flows and their requirements and the current route taken by each flow. Barring hardware failure, this is a perfect view of the network. The server utilizes that information in the computation of flow routes. Since it has access to current flow information about every link, it will, within the limits of its algorithm, always deliver a correct path. Once the route is determined, the server sends configuration messages to each router along the path and notifies the sender and

receiver of path establishment. Flow terminations are handled via the server as well. In case of server failure, a backup server may be present. This method does not require any changes to current routing protocols. If both servers fail, the network simply reverts to best-effort routing.

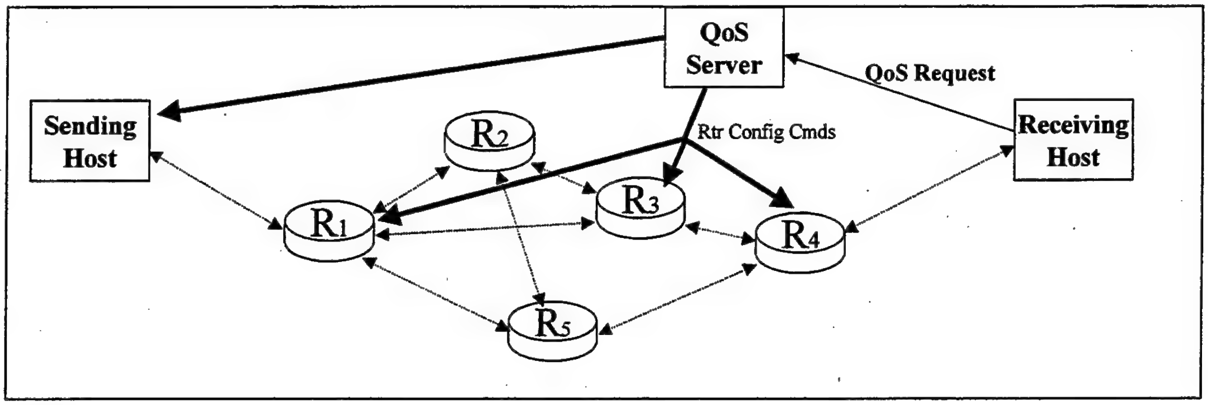

Figure 11: Alternative to RSVP Model

2. Advantages of the Server Approach

a. Works with existing routing protocols

RSVP requires the use of quality of service aware routing protocols. Under this model all route calculations are made at the centralized server, which then configures the affected routers. This model does not require the modification of existing routing protocols. Routers will only require changes to allow the server to configure parameters within the packet classifier.

b. Shift of Computational Complexity

Even in its simplest form, RSVP consumes router computational resources. Due to the increased complexity of routing algorithms, the requirement for each node to authenticate each host in the network and the tracking of thousands of simultaneous quality of service requests, the overhead of using RSVP is enormous. The server-based model of resource reservation shifts all of these burdens from each router in the network to the quality of service server. This frees the router to route, not compute paths.

Centralized path computation also reduces the total amount of calculations required through the network. Under the distributed model, each intermediate router must compute the best path algorithm. With centralized path discovery, the best-route algorithm is only computed once at the server.

c. Can Discover Alternate and Optimal Paths

Since the server model is not encumbered by the RSVP method of following a pre-determined route, it may consider alternate paths. In fact, because the global view of the server-based model considers all paths, there is no concept of alternate path. All paths are considered and once a route is determined, barring a hardware failure, it is guaranteed to work. If a link failure requires determination of a new path, the server calculates a new route.

A centralized QoS server can optimize the entire system. Obviously, this ability is a function of the complexity of the route-computing algorithms used on the

server. However, since route determination is no longer taking CPU time from the router, more computational time and more powerful processors may be dedicated to the problem. This increase in computational ability facilitates the use of complex algorithms which find not just an available path, but by considering all combinations of flows and link, arrive at optimized solutions.

d. *Routing protocol independent*

Unlike RSVP that requires use of a quality of service aware routing protocol, this model is truly independent of the network's routing protocol. Since the quality of service server analysis the entire system and configures routers to provide the service, the routing protocol used by the system is irrelevant. In fact, this model will work in the absence of a traditional routing protocol. There is no need to implement system wide complex quality of service routing protocols; for all that functionality is present in the server. Best effort traffic will continue to be routed via existing protocols.

e. *Centralization of Quality of Service Requirements*

Under RSVP, each router in the chain must be able to authenticate a host in the network before granting a request. This implies that each router must be aware of each host and their level of access. Under centralized quality of service, routers only need to authenticate the quality of service server. Once authenticated, the server is then authorized to allocate any resources. In turn, only the server need be of aware of hosts and their access levels.

THIS PAGE INTENTIONALLY LEFT BLANK

VI. CONCLUSIONS AND RECOMMENDATIONS

A. CONCLUSIONS

As a resource reservation agent, RSVP works as designed. However, the RSVP model is fundamentally flawed. RSVP's decentralized hop-by-hop route determination causes excessive network overhead and often arrives at erroneous path selection. The additional network overhead is caused by RSVP's reliance on sophisticated quality of service aware routing protocols to determine the path for a flow. The advanced routing protocols require the execution of complex algorithms on each intermediate node in the system, thereby reducing network performance. Incorrect path selection is inherent in the distributed model. Simply stated, the use of dated, non-real time information in path computation cause scenarios where the routing protocol selects a path that does not provide the required resources. Even if the routing protocol could correctly identify available paths, the RSVP model does not work because it requires the path of the flow to be determined before the quality of service parameters are known. This is an impossible task. Finally, since a decentralized routing protocol cannot compute optimal paths, by extension, the RSVP model cannot provide network optimization.

A simple solution to overcoming the deficiencies in the RSVP model lies in centralized computation of quality of service paths. Centralized computation of paths for quality of service flows offers the following advantages over the RSVP approach:

- Selected paths are guaranteed to be accurate.

- Routers are relieved of conducting optimal paths for flows.
- Network optimization algorithms may be used, for they are not competing for router computational resources.
- The centralized model is less complex than RSVP's model.

B. SUGGESTED FURTHER STUDIES

Prior to selection of an Internet standard for providing quality of service, several more issues need to be studied. Measurement, testing and simulation can further evaluate ideas presented in this thesis.

Research may be conducted to demonstrate the importance of IPv6 in quality of service enabled LANs. The test should center on increased network performance due to the increased efficiency of packet classification. Evaluation of RSVP under differing routing protocols would prove interesting. Exactly what is the difference in RSVP's performance when running alongside a quality of service routing protocol such as QOSPF versus a network running a standard, non-quality of service routing protocol?

LIST OF REFERENCES

[1] RFC 2212, *Specification of Guaranteed Quality of Service*, by Shenker, S., Partidge, C., and Guerin, R., September 1997.

[2] Micom Corporation, "Resource Reservation Protocol", [http://www.micom.com/WhitePapers/rsvp/ wprsvpte.htm]. May 1998.

[3] Goncalves, M. and Niles, K., *IPv6 Networks*, McGraw-Hill, February 1998.

[4] Armstrong, C. M, "Inflections Past & Future: New Directions for the Communications Industry.", [http://www.att.com/speeches/98/980604.maa.html]. June, 1998.

[5] Internet Request For Comment 1633, *Integrated Services in the Internet Architecture: an Overview,* Braden, R., Clark, D., and Shenker, S., June 1994.

[6] Internet Request For Comment 2205, *Resource SeSerVation Protocol (RSVP) -- Version 1 Functional Specification*, Braden, R., and others, September 1997.

[7] Internet Request For Comment 2676, *QoS Routing Mechanisms and OSPF Extensions*, Apostolopoulos, G., and others, August 1999.

THIS PAGE INTENTIONALLY LEFT BLANK

BIBLIOGRAPHY

Internet Draft Standard, *RSVP Extensions for IPv6 Flow Label Support*, Schmid, S., and others, August 1998.

Internet Request For Comment 2209, *Resource ReSerVation Protocol (RSVP) -- Version 1 Message Processing Rules*", Braden, R., and Zhang, L., September 1997.

Internet Request For Comment 2210, *The Use of RSVP with IETF Integrated Services*, Wroclawski, J., September 1997.

Internet Request For Comment 2212, *Specification of Guaranteed Quality of Service*, Shenker, S., Partridge, C., and Guerin, R., September 1997.

Internet Request For Comment 2386, *A Framework for QoS-based Routing in the Internet*, Crawley, E., August 1998.

Stallings, W., *Data and Computer Communications*, 5^{th} ed., Prentice-Hall, Inc., 1996.

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center 2
   8725 John J. Kingman Road, Ste 0944
   Fort Belvoir, VA 22060-6218

2. Dudley Knox Library 2
   Naval Postgraduate School
   411 Dyer Road
   Monterey, California 93943-5101

3. Director, Training and Education 1
   MCCDC, Code C46
   1019 Elliot Rd
   Quantico, VA 22134-5027

4. Director, Marine Corps Research Center 1
   MCCDC, Code C40RC
   2040 Broadway Street
   Quantico, VA 22134-5017

5. Director, Studies and Analysis Division 1
   MCCDC, Code C45
   300 Russell Road
   Quantico, VA 22134-5130

6. Professor Gilbert M. Lundy, Code CS 1
   Naval Postgraduate School
   Monterey, California 93943

7. Marine Corps Representative 1
   Naval Postgraduate School
   Code 037, Bldg 330, HA-220
   555 Dyer Road
   Monterey, California 93943

8. Marine Corps Tactical Systems Support Activity 1
   Technical Advisory Branch
   Attn: Maj J.C. Cummiskey
   Box 555171
   Camp Pendleton, CA 92055-5080